

Algoritmica – Esame di Laboratorio

13/06/2014

Istruzioni

Risolvete il seguente esercizio prestando particolare attenzione alla formattazione dell'input e dell'output. La correzione avverrà in maniera automatica eseguendo dei test e confrontando l'output prodotto dalla vostra soluzione con l'output atteso. Si ricorda che è possibile verificare la correttezza del vostro programma su un sottoinsieme dei input/output utilizzati. I file di input e output per i test sono nominati secondo lo schema: `input0.txt` `output0.txt` `input1.txt` `output1.txt` ... Per effettuare le vostre prove potete utilizzare il comando del terminale per la redirectione dell'input. Ad esempio

```
./compilato < input0.txt
```

effettua il test del vostro codice sui dati contenuti nel primo file di input, assumendo che `compilato` contenga la compilazione della vostra soluzione e che si trovi nella vostra home directory. Dovete aspettarvi che l'output coincida con quello contenuto nel file `output0.txt`. Per effettuare un controllo automatico sul primo file input `input0.txt` potete eseguire la sequenza di comandi

```
./compilato < input0.txt | diff - output0.txt
```

Questa esegue la vostra soluzione e controlla le differenze fra l'output prodotto e quello corretto.

Una volta consegnata, la vostra soluzione verrà valutata nel server di consegna utilizzando altri file di test non accessibili. Si ricorda di avvisare i docenti una volta che il server ha accettato una soluzione come corretta.

Esercizio

Dato un albero binario, definiamo “*visita sinistra*” dell’albero la visita che, partendo dalla radice, opera ricorsivamente come segue: se un nodo ha un figlio sinistro s , tale figlio è visitato e la visita prosegue da s ; altrimenti termina. Detto in altri termini, sono visitati tutti e soli i nodi presenti nel cammino che, partendo dalla radice, sceglie sempre e soltanto i figli sinistri, finché ce ne sono.

In maniera speculare, definiamo “*visita destra*” dell’albero la visita che, partendo dalla radice, opera ricorsivamente come segue: se un nodo ha un figlio destro d , tale figlio è visitato e la visita prosegue da d ; altrimenti termina.

Per ogni nodo u , consideriamo il sottoalbero radicato in u . Si definiscono:

- $L(u)$: il numero di nodi visitati dalla “visita sinistra” del sottoalbero, escluso u (zero se il nodo u è una foglia oppure se non ha un figlio sinistro);
- $R(u)$: il numero di nodi visitati dalla “visita destra” del sottoalbero, escluso u (zero se il nodo u è una foglia oppure se non ha un figlio destro);

Scrivere un programma che legga da tastiera una sequenza di N interi distinti e li inserisca in un albero binario di ricerca (senza ribilanciamento) nello stesso ordine con il quale vengono forniti in input. Il programma deve poi stampare in ordine crescente le chiavi dei nodi che soddisfano la condizione $L(u) > R(u)$.

NOTA: Affinché l’esame sia superato, la complessità in tempo dell’algoritmo **deve** essere lineare.

L’input è così formato:

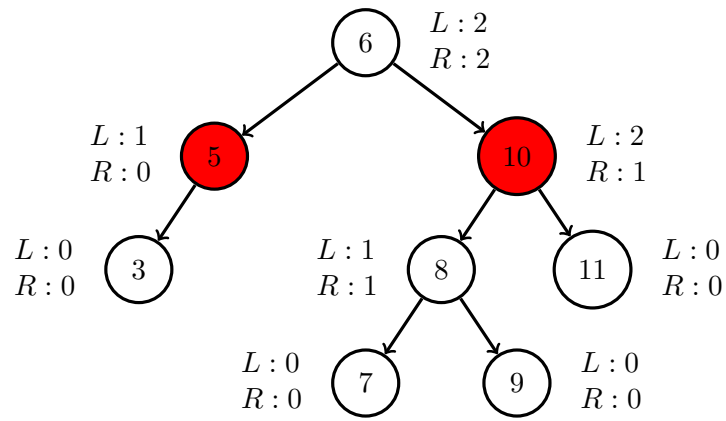
- la prima riga contiene il numero N di interi da inserire nell’albero binario di ricerca;
- le successive N righe contengono gli N interi, uno per riga.

L’output è costituito dalle chiavi dei nodi che soddisfano la condizione $L(u) > R(u)$. Tali chiavi vanno stampate in ordine crescente, una per riga.

Esempio

Input

8
6
5
3
10
8
7
9
11



Output

5
10

La figura dell'esempio mostra anche, per ogni nodo u , i valori di $L(u)$ e $R(u)$. In rosso si evidenziano i nodi u tali che $L(u) > R(u)$.